REPORT DOCUMENTATION PAGE	Form Approved OMB NO. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE Final Report	3. REPORT TYPE AND DATES COVERED 08/01/00 to 01/31/03

4. TITLE AND SUBTITLE Optimizing Heavily Loaded Agents	5. FUNDING NUMBERS DAAD190010484
6. AUTHOR(S) V.S. Subrahmanian	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Department of Computer Science University of Maryland A.V. Williams Building College Park, MD 20742	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) U. S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709-2211	10. SPONSORING / MONITORING AGENCY REPORT NUMBER 41418.1-CI

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12 a. DISTRIBUTION / AVAILABILITY STATEMENT	12 b. DISTRIBUTION CODE
Approved for public release; distribution unlimited.	

13. ABSTRACT (Maximum 200 words)

We develop algorithms to help scale software agents built on top of heterogeneous, legacy codebases. The algorithms apply to large data sets, to large volumes of workloads on agents, as well as algorithms for computationally intensive functions.

14. SUBJECT TERMS	15. NUMBER OF PAGES 7
	16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION ON THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev.2-89)
Prescribed by ANSI Std. 239-18
298-102

20030604 099

Final Report, May 10, 2003

Grant: Optimizing heavily loaded agents
Grant Number DAAD190010484
Army Research Office

Principal Investigator
V.S. Subrahmanian
Department of Computer Science A.V. Williams Building
University of Maryland
College Park, MD 20742.
Email: vs@cs.umd.edu
Tel: (301) 405-2711
Fax: (301) 405-8488

1 Scientific progress and accomplishments

With the increase in agent-based applications, there are now agent systems that support *concurrent* client accesses. The ability to process large volumes of simultaneous requests is critical in many such applications. In such a setting, the traditional approach of serving these requests one at a time via queues (e.g. FIFO queues, priority queues) is insufficient. Alternative models are essential to improve the performance of such *heavily loaded* agents. We have addressed the following technical problem. Suppose an agent is built on top of heterogeneous data structures (e.g. using methods such as those described in various agent framework).

> *Suppose the agent is confronted with a set S of requests. How should the agent process these requests so as to reduce the overall load on itself?*

For example, a powerpoint agent building PPT presentations for diverse military personnel may recognize the fact that many presentations requested by different clients require common financial data (or logistics data, or tactical data,....) to be computed and/or analyzed, and hence, performing this *once* instead of *many times* will most certainly enhance performance. Likewise, a diagnostic agent which merges multiple requests about how to repair or resupply parts on an M1-A1 tank engine will significantly reduce its load.

Contribution I: Merging methods Building on our previous contributions to the IMPACT (Interactive Maryland Platform for Agents Collaborating Together) Project, we highlight the following contributions made on heavily loaded agents.

- We developed an architecture through which an agent may merge requests;
- We developed the concept of an "invariant" that allows an agent developer to specify, implicitly, what it means for two jobs processed by an agent built on top of legacy software to be equivalent, contained, or overlapping.

- We developed provably sound and complete algorithms to compute, from such an implicit specification, all implied invariants.
- We developed algorithms that allow the agent to merge queries/jobs so as to reduce load on itself.
- We also developed heuristic algorithms for this purpose.
- We ran extensive experiments to determine the efficiency of our algorithms and concluded that the algorithms scale well.

Contribution II: Distribution algorithms The Internet contains a vast array of sources that provide identical or similar services. When an agent needs to solve a problem, it may split the problem into "subproblems" and find an agent to solve each of the subproblems. Later, it may combine the results of these subproblems to solve the original problem. In this case, the agent is faced with the task of determining to which agents to assign the subproblems. We call this the *agent selection problem* (ASP for short). Solving ASP is complex because it must take into account several different parameters. For instance, different agents might take different amounts of time to process a request. Different agents might provide varying "qualities" of answers. Network latencies associated with different agents might vary. In this work, we first formalized the agent selection problem and showed that it is NP-hard. We then proposed a generic cost function that is general enough to take into account the costs of (i) network and server loads, (ii) source computations, and (iii) internal mediator costs. We then developed exact and heuristic based algorithms to solve the agent selection problem.

Contribution III; Probabilistically Survivable MASs As multiagent systems (MASs) are increasing used for critical applications, the ability of these MASs to survive intact when various external events occur (e.g. power failures, OS crashes, etc.) becomes increasingly important. However, one never knows when and if a system will crash or be compromised, and hence, any model of MAS survivability must take this uncertainty into account.

We provide for the first time, a formal model for reasoning about survivability of MASs which includes both a declarative theory of survivability, as well as implemented algorithms to compute optimal ways of deploying MASs across a network.

A MAS-deployment specifies a placement of agents on various network nodes. Based on probabilistic information about the survivability of a given node, we develop a formal theory describing the probability that a given deployment will survive. This probability reflects the best guarantee we have of the MAS surviving. Our model does not assume that node failures are independent, though independence information can be easily added if so desired. The technical problem we need to grapple with is that of finding a MAS-deployment of the agents having the highest probability of survival. As we do not make unrealistic independence assumptions, this problem turns out to be intractable. As a consequence, heuristics are required to find a deployment (even if it is sub-optimal). We develop algorithms for the following tasks:

1. Given a MAS-deployment, how do we compute its probability of survival?

2. Find a MAS-deployment with the highest probability of survival - this algorithm is infeasible to implement in practice due to the above mentioned complexity results.

3. We develop a suite of heuristic algorithms to find (suboptimal) MAS-deployments.

We have conducted detailed experiments with our algorithm. Thes experiments show that our heuristic algorithms can find deployments very fast.

Contribution IV: Flexible authorization models Although several access control policies can be devised for controlling access to information, all existing authorization models, and the corresponding enforcement mechanisms, are based on a specific policy (usually the *closed* policy). As a consequence, although agents can make different policy choices in theory, in practice only a specific policy can be actually applied within a given system. We developed a unified framework that can enforce multiple access control policies within a single system. The framework is based on a language through which users can specify security policies to be enforced on specific accesses. The language allows the specification of both positive and negative authorizations and incorporates notions of authorization derivation, conflict resolution, and decision strategies. Different strategies may be applied to different users, groups, objects, or roles, based on the needs of the security policy. The overall result is a flexible and powerful, yet simple, framework which can easily capture many of the traditional access control policies as well as protection requirements that exist in real world applications, but are seldom supported by existing systems. The major advantage of our approach is that it can be used to specify different access control policies that can all coexist in the same system and be enforced by the same security server.

2 Technology transfer

In October 2002, we participated in a large Coalitions wargame (the demonstration was under DARPA funding, but used some technology developed under this ARO grant as well as some other sources) demonstration involving over 20 partners (including NRL, Lockheed Martin, BBN, etc.) where we track an enemy submarine has destroyed a coalition ship. Data from heterogeneous underwater sensors are used to predict where and the enemy submarine will be in the future. These predictions must take into account, not only the sensor readings, but also intelligence about the enemy sub's capabilities, terrain information (as the sub cannot travel on land!), ocean depth data. Based on the large space of possible predictions, the system must make the following decisions. Who will intercept the enemy? Where will the intercept occur? When will it occur? Should multiple intercept points be planned for? We plan to extend our coalitions demonstration to answer these questions. This demonstration was given in October 2002 to RADM Ronald Route at the Naval Warfare Development Command in Rhode Island. Our component of this demonstration worked very well and showed how our techniques can be effectively used in military operations.

Jointly with SAIC and the Maj. John McKitrick of the Chief Technology Office, DISC4, United States Army, we jointly built an application called SMART which integrates a set of Army logistics databases (SORTS and AFM). Users of SORTS and AFM currently have to wait several days before they receive updates. Integrated access to these data sources is

nonexistent. Thus users get only a partial view of stale data. To integrate the data, they must access both individually and merge the results manually. Furthermore, they must wait till fresh data is available. The goal of SMART is to

- provide a single framework within which to seamlessly access SORTS and AFM data
- provide users the ability to slice and dice logistics data in many different ways
- develop techniques for agents to scale to the data sizes involved and validate these methods on large data sets
- develop methods to automatically visualize the resulting data
- develop methods for agent integration to deal with DoD firewalls without compromising the integrity of the firewall.

We believe that our effort with SMART highlighted our IMPACT system that involves many of the techniques described above. Specifically, the following observations were made.

- IMPACT's CIOS (client interface object server) made it very easy for SAIC's graphical user interfaces to access IMPACT agents and graphically present the results to the user.
- IMPACT's Agent Development Environment provided JDBC connectors that enabled the SAIC-UMD team to easily integrate Oracle, Access and Sybase data. Even though neither AFM nor SORTS is in ACCESS, some of the sample data was integrated using ACCESS during the interim phases of SMART development.
- IMPACT provided tools and techniques that allowed the SAIC-UMD team to easily maneuver distributed computations that needed to get through an Army firewall. The ability to easily work with firewall principles and still protect the integrity of the firewall was a major accomplishment of the joint effort.
- IMPACT agents provided various value-added services. They were able to slice and dice ARMY logistics (SORTS and AFM) data in many different ways (by year, by unit, etc.) and present them in appropriate ways to end users.
- The IMPACT agents thus built scaled very well to the quantity of data involved, despite the computational overhead incurred in distributed data access and firewall access.
- The use of IMPACT made the SMART system very flexible. Changing the behavior of agents is very easy (and will hold for future changes as well). Adding new agents into an integrated agent environment was also very easy - this makes SMART flexible because as we expand SMART to include new data sources and new workflows, the use of IMPACT agents makes such expansions very easy.
- Building the SMART application using IMPACT took much less time than it would have otherwise.

3 Students

F. Ozcan (PhD - now at IBM Almaden Research Center), M. Fayzullin, Robert Ross (PhD expected in 2003).

4 Publications

The following papers benefited from the funding provided by ARO.

1. J. Dix, S. Kraus and V.S. Subrahmanian. Temporal Agent Programs, ARTIFICIAL INTELLIGENCE journal. Vol. 127, 1, pps 87–135, March 2001. i

2. E. Hwang and V.S. Subrahmanian Presentation Planning for Distributed Video Systems. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, Vol. 14, 5, pps 1059-1077, Sep/Oct 2002.

3. A. Dekhtyar, R. Ross, and V.S. Subrahmanian. Probabilistic Temporal Databases, I, ACM Transactions on Database Systems (TODS), Vol 26, Nr. 1, March 2001.

4. S. Jajodia, P. Samarati, M.L. Sapino and V.S. Subrahmanian. Flexible Support for Multiple Access Control Policies, ACM Transactions on Database Systems (TODS) Vol. 26, Nr. 2, pps. 214 - 260, June 2001.

5. 1. T. Eiter, J. Lu, T. Lukasiewicz and V.S. Subrahmanian. Probabilistic Object Bases, ACM Transactions on Database Systems, Sep. 2001.

6. P. Bonatti, S. Kraus and V.S. Subrahmanian Secure Agents, Annals of Math and Artificial Intelligence Journal, 37 $(1-2)$, pages 169-235, 2003.

7. V. Biazzo, R. Giugno, T. Lukasiewicz and V.S. Subrahmanian. Temporal Prob abilistic Object Bases. Accepted for publication in IEEE Transactions on Knowledge and Data Engineering.

8. N. Leone, F. Scarcello and V.S. Subrahmanian. Optimal Models of Disjunctive Logic Programs: Semantics, Complexity, and Computation. Accepted for publication in IEEE Transactions on Knowledge and Data Engineering.

9. A. Dekhtyar, F. Ozcan, R. Ross and V.S. Subrahmanian. Probabilistic Temporal Databases, II: Calculus and Query Processing. Manuscript.

10. T. Eiter, V. Mascardi and V.S. Subrahmanian. Error-Tolerant Agents, Computational Logic: Logic Programming and Beyond, (eds. F. Sadri and A. Kakas), Springer-Verlag, 2002, pages 586-625.

11. R. Ross, V.S. Subrahmanian, J. Grant. Probabilistic aggregates. Proc. ISMIS 2002, pages 553-564 (Springer Lecture Notes in AI).

12. M. Fayzullin and V.S. Subrahmanian. An algebra for Powerpoint sources, Intl. Workshop on Multimedia Information Systems, pages 155-164, 2002.